目录

[一、晶圆类型（WaferType） 3](#_Toc106375460)

[1.界面 3](#_Toc106375461)

[2.功能 3](#_Toc106375462)

[2.1 入口 3](#_Toc106375463)

[2.2 功能介绍 3](#_Toc106375464)

[3.代码 3](#_Toc106375465)

[3.1 项目位置 3](#_Toc106375466)

[3.2 使用的类 4](#_Toc106375467)

[3.3 开发过程 5](#_Toc106375468)

[二、JOB管理 6](#_Toc106375469)

[1.界面 6](#_Toc106375470)

[2.功能 6](#_Toc106375471)

[2.1 入口 6](#_Toc106375472)

[2.2 功能介绍 7](#_Toc106375473)

[2.3 界面逻辑及注意事项 7](#_Toc106375474)

[3 代码 7](#_Toc106375475)

[3.1项目位置 7](#_Toc106375476)

[3.2 使用的类 7](#_Toc106375477)

[3.3 开发过程 8](#_Toc106375478)

[三、程式主界面 9](#_Toc106375479)

[1.界面 9](#_Toc106375480)

[2.功能 9](#_Toc106375481)

[2.1 入口 9](#_Toc106375482)

[2.2 功能介绍 9](#_Toc106375483)

[2.3 界面逻辑及注意事项 11](#_Toc106375484)

[3.代码分析 12](#_Toc106375485)

[3.1 Bach.Job.FineAlign\_Mark\_NoDB 12](#_Toc106375486)

[3.2 Bach.Job.Die 15](#_Toc106375487)

[3.3 Bach.Job.PassiveFocus 15](#_Toc106375488)

[3.4 Bach.Job.Inspection 16](#_Toc106375489)

[3.5 Bach.Job.Map 16](#_Toc106375490)

[3.6 Bach.Job.Measure 17](#_Toc106375491)

[3.7 Bach.Job.UnDefine 19](#_Toc106375492)

[3.8 Picasso.UI.WPF 19](#_Toc106375493)

[3.9 Bach.Job.Wafer 19](#_Toc106375494)

[3.10 Bach.Job.WaferNoGraphic 20](#_Toc106375495)

[四、V4版本改造内容 20](#_Toc106375496)

[界面WPF改造 20](#_Toc106375497)

[检测界面拆分，检测核心类拆分 20](#_Toc106375498)

[代码精简合并 20](#_Toc106375499)

# 一、晶圆类型（WaferType）

## 1.界面



## 2.功能

### 2.1 入口

【菜单栏】-【检测和设置】-【WaferType】

### 2.2 功能介绍

界面包含晶圆所属一些基本参数，包含接片类型、缺边缺口、是否无图形光板、是否3D晶圆等等。在此预定义类型，可以减少后续制作程式工作量。

## 3.代码

### 3.1 项目位置

System.Settings.WaferType

### 3.2 使用的类

#### 3.2.1 System.Settings.WaferType.WaferType（晶圆类型数据实体）

包含以下属性：

##### Handling（接片类型）

包括纯晶圆、带孔晶圆以及带框晶圆。

用以区分使用牙叉的类型，像带框晶圆（一般指蓝膜晶圆）使用的牙叉和纯晶圆的就不能通用。

注：该参数暂未使用。

##### Transparency（透明度）

包括不透明、半透明、以及透明。

用以区分晶圆的透明度，不同透明度可以使用的检测方式不同。例如：不透明晶圆，那就无法使用透射光检测等等，后期会在制作程式中选择检测光源时加以判断。

注：此参数暂未使用。

##### FiducialType（缺口类型）

包含一条缺边，两条缺边以及缺口。

用以区分晶圆的缺口类型，在预对准时会使用到该参数，不同类型使用的预对准的配置不同。

注：此参数暂未使用。

##### UsePrealigner（启用预对准）ReturnViaPrealigner（启用放回对准）

这两个参数都使用到预对准硬件，前者是从晶圆盒取出后，执行预对准，再放置到平台；后者是从平台取片后，执行对准，再放回晶圆盒。

注：此参数未使用

##### UseWaitStation（启动等待站点）

启用此参数，交换片时，会执行等待。

##### ScanWaferId（是否扫描WaferId）

##### MainJobType（晶圆检测类型 ）

分为Pattern（有图形检测）和UnPattern（无图形检测）。

不同检测类型对界面的展示以及算法的选择都有影响。

##### Use3D（启用3D）

使用启用3D检测。

### 3.3 开发过程

数据实体中原本包含形状和尺寸两个参数，在后来的设计中，这两个参数被迁移到晶圆结构（AT.Exposure.Setups.SetupLayout）数据实体中，这两个参数和晶圆结构的相关性比较大。形状确定晶圆结构边框，尺寸和Die尺寸同时确定了Die的数量，且这两个参数会经常改动，故迁移至晶圆结构实体类型。

界面UI原本计划使用属性框（PropertyGrid）承载数据，前台改动能够实时同步到至后台数据，且属性的各项Attribute也能够对属性进行很好的分类。但是，由于参数数量不是很多，属性框和风格和V4版本风格不契合，故而采用WPF+MVVM开发了仿表单风格的界面UI。

# 二、JOB管理

## 1.界面



## 2.功能

### 2.1 入口

【菜单栏】-【功能】-【JOB】

### 2.2 功能介绍

界面主要功能是管理创建的JOB，包含 创建、删除、重命名等功能。界面分为三个模块【选择JOB】、【Setup】和【Recipes】。

* 【选择JOB】模块用于创建JOB以及管理现有JOB；
* 【Setup】模块显示当前选中JOB的晶圆类型（WaferType），在此可以修改JOB所属晶圆类型；
* 【Recipes】模块展示了当前JOB选择的Recipe；

### 2.3 界面逻辑及注意事项

* 如果当前JOB已经打开，将无法删除以及重命名（JOB文件被占用），可以选择其他JOB打开后再执行操作；
* 【Setup】模块部分参数修改后，程式部分步骤将需要重新制作，才能进行检测；

## 3 代码

### 3.1项目位置

Picasso.UI.WPF.View. FormSelectJob

### 3.2 使用的类

#### 3.2.1 Picasso.UI.WPF.View.Itree

树结构接口，包含Items（子节点）和TreeParent（父节点）属性

#### 3.2.2 Picasso.UI.WPF.View.TreeItem

树节点实体类型继承Itree接口，与UI绑定，控制UI显示以及交互，包括节点展开（Expand），收起（Collapsed），显示隐藏，选中，复制，粘贴、删除、重命名，双击事件，点击事件等。

#### 3.2.3 AT.Exposure.Setups.JtreeNode

包含NodeText（节点名称）、NodeType（节点类型：文件/文件夹），NodeTag（节点对应数据），Items（子节点），ID（节点GUID），ParentID（父节点ID）等属性。

用于界面树结构的存储和获取的中间类型。

#### 3.2.4 AT.Exposure.Setups.ProductJobHelper

JOB管理数据库帮助类，用于将界面树结构保存到数据库以及从数据库取出树结构更新到UI。

### 3.3 开发过程

 V3版本使用.NET自带Winform的TreeView构建树结构，V4中整体使用WPF后，原计划使用WPF.TreeView设计界面。但是，由于WPF.TreeView相对Winform.TreeView缺少许多内置方法，并且UI并不好看，所以采用WPF.ItemsControl + MVVM重新创建树结构界面，并根据需要添加各种常用功能。目前常用功能已增加，缺少拖动节点改变层级的交互方法。

# 三、程式主界面

## 1.界面



## 2.功能

### 2.1 入口

【菜单栏】-【功能】-【JOB】-【打开】

### 2.2 功能介绍

界面自上而下可以分为4个部分：基础设置、进阶设置、Recipes管理以及Map展示。

#### 2.2.1 基础设置界面

界面是根据选择的晶圆类型（Pattern，Unpattern，启用3D）、形状（Circle，Rectangle、None）等自动生成，以此减少部分无用参数的干扰，简化程式的制作。

部分参数（圆中心、精细计算、创建）该界面只做展示，具体计算步骤需要点击按钮进入对应界面再执行计算。这样设计，可以将参数显示和计算分隔开，也使得主界面更加简洁明了。

#### 2.2.2 进阶设置

界面根据选择的晶圆类型（Pattern和UnPattern）判断不同类型使用到的检测模块自动生成。这样便于后期增加、减少检测模块，也使得各个模块之间分隔开，开发人员可以各自负责各自的功能模块。

每个步骤使用颜色区分是否只做完成，未完成为黑色，完成后变为绿色，当部分关键检测参数（检测模式：前道、后道等）改变后，会重置所有步骤状态，以此提示用户重新只做程式。

#### 2.2.3 Recipe管理

该界面进入后可以创建、修改、删除Recipe模板，可以选择当前JOB使用的Recipe。检测中可以根据选择的Recipe对一张晶圆片执行多次不同参数的检测。

#### 2.2.4 Map展示

该界面展示根据形状、Die尺寸、外围尺寸、晶圆大小等生成的晶圆结构，也用作检测中显示检测进度、缺陷位置以及缺陷分布。该界面基于WPF技术开发，在Die个数以及缺陷个数很多的情况下，也能获取更好的使用体验。

### 2.3 界面逻辑及注意事项

* 【圆中心】（标记左上角点、标记右下角）会根据选择的形状控制显示隐藏。
* 【圆中心】（标记左上角点、标记右下角），【精细计算】以及【创建结构】初始状态为红色，计算完成后变为绿色；
* 【检测模式】只有在晶圆类型为【Pattern】且【不启用3D】的情况下才会显示。
* 【检测模式】修改后，进阶设置所有模块步骤状态将被改为未完成，字体颜色变为黑色。
* 【形状】【尺寸】【Die尺寸】【外围尺寸】【结构模式】【边缘】改变后，右侧会提示重置结构，点击重置后才会变更晶圆结构，避免误操作导致结构的变化。
* 当【形状】选择为【None】时，只能通过执行【Map导入】改变晶圆结构；当【形状】【不为None】且执行【Map导入后】，【形状】将被自动变更为【None】。

## 3.代码分析

### 3.1 Bach.Job.FineAlign\_Mark\_NoDB

对准模块项目，包含：

对准JOB核心类：Bach.Job.FineAlign\_Mark\_NoDB.Business.FineAlignJob

#### 使用Setup\_pre\_view()方法注册对准进阶设置界面

**

这里注册的页面，在程式主界面加载时会被填充到基础设置区域中，如果为null，则不填充。

#### 使用Setup\_view()方法注册对准进阶设置界面

**

这里注册的页面，在程式主界面加载时会被填充到进阶设置区域中，如果为null，则不填充。

#### 使用Handle(IHandleItem item)方法执行检测中步骤

**

这里执行[检测中步骤](#_构造函数中初始化有图形检测涉及到的主要步骤：)对应的方法。

程式制作对准界面：Bach.Job.FineAlign\_Mark\_NoDB.WPFUI.UCAlignment

对准方法、根据Die行列定位Die实际位置等等一些方法的核心类：Bach.Job.FineAlign\_Mark\_NoDB.Business.MarkAlignment

检测中涉及到对准的相关方法核心类：

Bach.Job.FineAlign\_Mark\_NoDB.Business. InspectingAlignment

### 3.2 Bach.Job.Die

Die计算模块项目，包含：

DieJOB核心类：Bach.Job.Die.Business.DieJob

类型结构与[3.1](#_使用Setup_pre_view()方法注册对准进阶设置界面)类似

采图、扫描晶圆结构等方法核心类：

Bach.Job.Die.Core.CaptureDieImg

Bach.Job.Die.Core.CreateMappingOption

Bach.Job.Die.Core.DieScanForMapping

主界面：Bach.Job.Die.WPFUI.UCSetupMain



圆中心计算界面：Bach.Job.Die.WPFUI.UCCalculateCircelCenter

Die尺寸精细计算界面：Bach.Job.Die.WPFUI.UCDieAccurate

扫描Map图界面：Bach.Job.Die.WPFUI. UCScanMap

### 3.3 Bach.Job.PassiveFocus

拟合聚焦功能模块项目，包含：

拟合聚焦JOB核心类：Bach.Job.PassiveFocus.Business.PassiveFocusJob

聚焦方法类：Bach.Job.PassiveFocus.Business. PassiveFocusImp

检测中拟合聚焦方法类：

Bach.Job.PassiveFocus.Business. InspectingPassiveFocus

程式制作拟合聚焦界面：

Bach.Job.PassiveFocus.WPFUI.UCPassiveFocus

### 3.4 Bach.Job.Inspection

检测相关项目，包括：

检测JOB核心类：Bach.Job.Inspection.Business.InspectionJob

类型结构与[*3.1*](#_使用Setup_pre_view()方法注册对准进阶设置界面)类似

检测Die行列划分方法类：*Bach.Job.Inspection.Business.BaseInspection*

扫描条带相关计算方法类(包含2D检测和3D检测)：

*Bach.Job.Inspection.Business.* *StripHandler2D*

*Bach.Job.Inspection.Business.* *StripHandler3D*

程式制作检测参数界面：*Bach.Job.Inspection.WPFUI.UCInspectParamSetting*

### 3.5 Bach.Job.Map

Map图显示相关项目。

MapJOB核心类：Bach.Job.Map.Bussiness.MapJob

该JOB初始化过程中，会更新界面Map图：



其余类型结构与[3.1](#_使用Setup_pre_view()方法注册对准进阶设置界面)类似

### 3.6 Bach.Job.Measure

算法设置相关项目，包括Golden图采集，编辑，检测中初始化算法参数等。

算法JOB核心类：Bach.Job.Measure.Business.MeasureJob

Golden采集以及编辑相关类型：

*Bach.Job.Measure.Business.CaptureDieImg*

*Bach.Job.Measure.Business*.*MultiStrips\_CaptureDieImg*

*Bach.Job.Measure.Business*.*MultiStripsInspectGoldenSetting*

*Bach.Job.Measure.Business*.*SetupInspectGoldenSetting*

*Bach.Job.Measure.Business*.*StripsInspectGoldenSetting*

初始化算法侧资源类：

*Bach.Job.Measure.Business.CoreResAssist*

检测中和算法相关方法类：

*Bach.Job.Measure.Business*.*InspectingMeasure*

Goldend图采集界面：*Bach.Job.Measure.WPFUI*.*UCFormSelectDie*



Golend图编辑界面：*Bach.Job.Measure.WPFUI.* *MultiGoldenMap* 

### 3.7 Bach.Job.UnDefine

这个项目包含一些检测中涉及到的操作，这些操作和对准、聚焦、算法等等相关性不大，例如：初始化检测任务，程式校验等等。

UnDefineJOB核心类：Bach.Job.UnDefine.Business. UnDefineJob

### 3.8 Picasso.UI.WPF

这个项目时V4版本改造最多的核心项目，包含V4版本界面框架，功能页面，图标图片资源等等。

具体内容过多，详细可以看代码。

### 3.9 Bach.Job.Wafer

有图形检测MainJOB项目。

核心类型：Bach.Job.Wafer.Business.WaferJob

#### 构造函数中初始化有图形检测涉及到的主要步骤：

检测开始前流程：



该步骤在每次检测中只会执行一次，包含子JOB校验、运行参数、设置Golden图等等。

检测中针对每一片流程：



该步骤针对每一片Wafer都会执行。

检测结束流程：



以上每个步骤流程的相关代码被拆分到 [3.1](#_3.1_Bach.Job.FineAlign_Mark_NoDB)至[3.7](#_3.7_Bach.Job.UnDefine) 的每一个子JOB中，在此增减步骤，可以控制检测的流程。

#### Config(MainJobConfig mainJobConfig)方法注册程式制作流程



在程式制作界面初始化过程中，会从代码中获取有图形检测注册的所有子JOB，使用JOB.GetSetupView()和JOB.GetPreSetupView()方法取出相关界面，填充到主页面上，以此来控制不同检测类型程式制作的界面显示。

### 3.10 Bach.Job.WaferNoGraphic

无图形检测MainJOB项目。

核心类型：Bach.Job.WaferNoGraphic.Business

其他项目结构和 [3.9：有图形检测MainJOB项目一致](#_3.9_Bach.Job.Wafer)

# 四、V4版本改造内容

## 界面WPF改造

绝大多数界面都使用WPF技术重写，除个别：相机界面，Map图界面等。相较于V3，界面交互更加流畅，前后台数据交互、界面控制更加效率。

## 检测界面拆分，检测核心类拆分

V3版中的检测界面AT.Workflow.WinForms.FormInspectWindow，检测核心类AT.Workflow.Workers.Inspect.D2DscanningInspectWorker 承载了过多功能。在这两个类型中增加新功能或者定位修改BUG都是对开发人员的极大考验。

V4中，这两个类型的相关功能被拆分到各自子JOB中，例如：[检测Die行列及条带计算](#_3.4_Bach.Job.Inspection)、[检测中对准及起始点定位](#_3.1_Bach.Job.FineAlign_Mark_NoDB)、[检测前初始化资源](#_3.6_Bach.Job.Measure)等等，项目结构更加清晰，增加修改功能也更加方便。

## 代码精简合并

V3中规划扫描路径相关代码：



主要分为圆形前道、圆形后道、矩形前道、矩形后道，类型名称有歧义；

代码中重复的字段声明，重复方法也有许多，改动起来不方便。

V4中拆分为一个基类和4个子类：



名称命名更加准确；公用方法、字段提取到基类中，代码更加简洁。